

# Avalon Transcoding API

The Transcoding API will consist of a set of lightweight Ruby classes that will be able to initiate transcoding jobs, distribute derivative files, track/poll job status, and perform common maintenance tasks such as prioritization, cancellation, and cleanup.

## Definitions

Consumer	Application code that makes use of the Transcoding API.
Implementation	The set of classes and methods that make a given Transcoder available to a Consumer via the methods defined in the Transcoding API
Job	A single piece of transcoding work
Transcoder	An implementation-specific provider of transcoding services. Examples include OpenCast Matterhorn, Kaltura, Zencoder, and Amazon Elastic Transcoder.
Transcoding API	The set of classes and methods that can be used to initiate, track and manipulate transcoding jobs.

## Classes

### Transcoder::Base

An abstract model that serves as the reference for an Implementation.

#### Class Methods

```
create(input, output, opts={})
```

Begin a transcoding operation. Returns a job.

```
find(job_id)
```

Load and return a single job.

```
find_by_status(*states)
```

Find all jobs currently in the given state(s).

#### Instance attributes

- `job_id`
- `state`
- `current_operations`
- `errors`
- `original_filename`
- `tech_metadata`

#### Instance Methods

- cancel!
- canceled?
- complete?
- purge!
- running?
- update!

## Chris's F2F Notes

### File submission

- Select (or use default?) preset/settings
- Generate MediaObject ID
- Save MediaObject

Move file to uploads directory (collection/uploads?) master\_file/create

Characterize

### Process (transcode)

#### Create job

##### Submit

- Input file (URL?)
- Output destination (URL?)
- Encoding options

##### Return

- Job ID

### Background / Encoder

#### Poll (status?)

##### Submit

- Job ID

##### Return

- State
- %
- Current operations?

#### Finished (Details)

##### Submit

- Job ID

## Return

- Status
- %
- URL
- Technical metadata

## Failed (Details)

### Submit

- Job ID

### Return

- Status
- %
- Error messages
- Current operations?

*Depend on encoder/encoder shim to transfer derivatives to streaming server/location*

## Chris's LCDX Notes

### Basic Requirements for local encoding service

- Start encoding job
- status of encoding job
  - succeed/failed/canceled
  - current operation (could be hash due to parallelization of derivative creation)
  - errors
  - process complete
- further in depth details of encoding job (media info report)
  - techMD (aspect ratio, framesize, bitrate)
  - derivatives created (id, url/label, hash)
  - id
  - original filename

### Proposed Operation

Shared code, API layer for work. Run by institution. Avalon wants local FFMPEG queue based solution.

1. Submit job, ID is returned
2. Use ID for status query
3. Derivatives placed in proper location by service

### Other Proposed Features

1. Cancel job and retract derivatives (time intensive and some services charge per minute)
2. Need to support priority flagging, not present in some codebases

## Avalon

## Jobs

### Create

#### Submit

Input file

Output file

Encoder settings

#### Response

Job ID

### Details

#### Submit

Job ID

#### Response

Status

Progress %

Current operations

URL

Technical metadata (input file/output files)

Errors

### List by Status

#### Submit

Job ID

#### Response

Status

Progress %

Current operations

URL

Technical metadata (input file/output files)

Errors

### Cancel

#### Submit

Job ID

## Response

Cancellation confirmation

## Retract Derivatives

### Submit

Job ID

## Response

Retraction confirmation

# Zencoder

Zencoder uses a 'Job' to transcode media. The Job specifies all settings in a single package.

API versioned by version number, e.g. 'v2'

Each Job, Input and Output is assigned an (independent) ID

Job states include pending, waiting, processing, finished, failed, and cancelled.

Input states include pending, waiting, processing, finished, failed, and cancelled.

Output states include waiting, queued, assigning, processing, finished, failed, cancelled and no input.

[Error codes](#)

## Jobs

### Create

### Submit

Input

Encoder settings (optional)

## Response

Job ID

Output ID

### List

### Submit

## Response

Job details for each job

## Details

## Submit

Job ID

## Response

Job details

- Created at
- Finished at
- Updated at
- Submitted at
- Pass-through
- Job ID

Input details

- Input ID
- Format
- Framerate
- Height
- Width
- etc.

Output details

- Output ID
- Format
- Framerate
- Height
- Width
- etc.

State

## Resubmit

## Submit

## Response

## Cancel

## Submit

## Response

## Finish (a Live Job)

## Submit

## Response

## Job Progress

## Submit

Job ID

## Response

Job state

Job progress

Input state

Input progress

Output state

Output progress

## Notifications

Register for a notification when a job is completed via HTTP POST, email, etc.

### Returns:

- Job ID
- Status: 'finished', 'failed', 'cancelled'
- Output label (if applicable)

## Amazon Elastic Encoder

Amazon uses a 'Pipeline' (queue) to transcode a 'Job' according to the settings specified in a 'Preset'.

API versioned by date, e.g. '2012-09-25'

The value of `Job:Status` is one of the following: `Submitted`, `Progressing`, `Complete`, `Canceled`, or `Error`

Errors

## Jobs

### Create

#### Submit

Input file

Output file

Preset ID (encoder settings)

### Response

Job details

Input settings

Output details

### List by Pipeline

#### Submit

### Response

## List by Status

Submit

Response

Read

Submit

Job ID

## Response

Input details

- File
- Frame rate
- Resolution
- Aspect ratio
- Interlaced
- Container
- Detected properties (width, height, frame rate, file size, duration)

Output details

- ID
- Output file
- Rotate
- Preset ID
- Duration
- Width
- Height
- Frame rate
- File size
- Status
- etc.

Job details

- User metadata
- Status
- Timing (submit, start, finish)

Cancel

Submit

Response

## Notifications

Register for a notification when a Job changes state via HTTP, email, etc.

Returns:

- State: 'PROGRESSING', 'COMPLETED', 'WARNING', 'ERROR'
- Error code



- Message
- Version (API)
- Job ID
- Pipeline ID
- Input